1.   (Previously presented) A method for determining dependencies between a first graphics primitive and a second graphics primitive, the method comprising:

calculating a first bounding box for the first graphics primitive, the first bounding box surrounding at least one source operand of the first graphics primitive;

calculating a second bounding box for the second graphic primitive, the second bounding box surrounding a destination operand of the second graphic primitive; and

determining whether the first bounding box and the second bounding box overlap wherein a dependency is detected if the bounding boxes overlap.

2.   (Cancelled)

3.   (Cancelled)

4.   (Previously presented) The method of claim 1, wherein a write after read dependency is detected if the second bounding box overlaps the first bounding box.

5.   (Cancelled)

6.   (Cancelled)

7.   (Currently amended) The method of claim 1 wherein a read after write dependency is detected if the first bounding box overlaps the second bounding box, <u>where the second bounding box surrounds an earlier primitive.</u>

8.   (Currently amended) A method for determining whether a dependency exists between a first graphics primitive and a second graphics primitive, comprising:

comparing a set of destination pixel locations of the first graphics primitive with a set of destination pixel locations of the second graphics primitive; and

determining whether a <u>write after write</u> dependency exists between the first and second graphics primitives as a function of the comparison of the destination pixel locations of the first and second graphic primitive.

9.   (Previously presented) The method of claim 8 wherein the step of comparing further comprises:

calculating a first bounding box which surrounds the set of destination pixel locations of first graphics primitive;

calculating a second bounding box which surrounds the set of destination pixel locations of the second graphics primitive; and

comparing the first bounding box with the second bounding box.

10.   (Cancelled)

11.   (Previously presented) A method for determining whether a dependency exists between a first graphics primitive and a second graphics primitive, comprising:

comparing a set of destination pixel locations of the first graphics primitive with at least one set of source pixel locations of the second graphics primitive; and

determining whether a dependency exists between the first and second graphics primitives as a function of the comparison.

12.   (Previously presented) The method of claim 11 wherein the step of comparing further comprises:

     calculating a first bounding box which surrounds the set of destination pixel locations of the first graphics primitive;

     calculating a second bounding box for each of the at least one set of source pixel locations of the second graphics primitive; and

     determining whether there is dependency if the first and second bounding boxes overlap.

13.   (Currently amended) The ~~process~~ <u>method</u> of claim 12, wherein a dependency exists between the first and second graphic primitive if the first and second bounding boxes overlap.

14.   (Previously presented)   An apparatus for detecting dependencies between a first graphics primitive and a second graphics primitive, comprising:

     a destination reservation station for storing a destination bounding box location for the first graphics primitive;

     a source reservation station for storing a source bounding box location for the first graphics primitive; and

     a first comparator for comparing the destination bounding box location for the first graphics primitive with a bounding box location of the second graphics primitive and generating a first resultant bit.

15.   (Previously presented) The apparatus of claim 14 further comprising:

     a second comparator for comparing the source bounding box location for the first graphics primitive with a destination bounding box location of the second graphics primitive and generating a second resultant bit.

16. (Previously presented) The apparatus of claim 15 further comprising:

a third comparator for comparing the destination bounding box location for the first graphics primitive with a source bounding box location of the second graphics primitive and generating a third resultant bit.

17. (Original) The apparatus of claim 16, further comprising:

a logic OR gate for receiving the first, second, and third resultant bits and performing a logical OR operation in order to determine whether any dependencies exist between the first and second graphics primitives.

18. (Original) A method for parallel processing of a plurality of 3D primitives in an out of order sequence comprising:

storing the plurality of 3D primitives in a queue;

processing at least two of the primitives in order at the same time, where that at least two of the primitives have no dependency and wherein a first primitive in the at least two is completely processed before the others;

detecting a dependency between a next primitive to be processed from the plurality in the queue and the primitives in the at least two which have not yet been completely processed; and

skipping the next primitive to be processed from the plurality in the queue and processing a subsequent primitive from the plurality in the queue, wherein no dependency is detected between the subsequent primitive and the primitives in the at least two which have not yet been completely processed.

19. (Original) The method of claim 18 where the step of detecting is comprised of:

comparing destination regions for the primitives in the at least two which have not yet been completely processed with a new destination region and source regions for the next primitive to be processed in order to detect any overlap; and

detecting a dependency where there is an overlap between the destination regions for the primitives in the at least two which have not yet completely processed and either the new destination region or the source regions for the next primitive to be processed.

20. (Original) A method for providing out of order processing of a plurality of 3D primitives comprising:

storing the plurality of 3D primitives, including source and destination region bounding box coordinates for each primitive, in an issue unit;

processing at least two of the primitives at the same time, where that at least two of the primitives have no dependency; and

detecting a dependency between a next primitive to be processed from the plurality in the issue unit and the primitives in the at least two which have not yet been completely processed, wherein the next primitive to be processed from the plurality in the issue unit is skipped if a dependency is detected.

21. (Original) The method of claim 20, wherein a subsequent primitive form the plurality in the issue unit is processed if no dependency is detected between the subsequent primitive and the primitives in the at least two which have not yet been completely processed.

22.   (Previously presented) The method of claim 20 wherein the step of detecting is comprised of:

comparing the destination region bounding box coordinates for the primitives in the at least two which have not yet been completely processed with the destination region bounding box coordinates and the source region bounding box coordinates for the next primitive to be processed in order to detect a dependency; and

detecting a dependency if there is an overlap in the destination region bounding box coordinates for the primitives in the at least two which have not yet been completely processed and either the destination region bounding box coordinates or the source region bounding box coordinates for the next primitive to be processed.

23.   (Original) An apparatus for providing out of order processing of a plurality of graphics primitives comprising:

an issue unit for storing the plurality of graphics primitives including source and destination region bounding box coordinates for each primitive in the plurality;

two or more accelerators for processing at least two of the primitives in the plurality at the same time, where the at least two of the primitives have no dependency; and

comparators for detecting a dependency between a next primitive to be processed from the plurality in the issue unit and the primitives in the at least two which have not yet been completely processed, wherein the next primitive to be processed from the plurality in the issue unit is skipped if a dependency is detected.

24.   (Original) The apparatus of claim 23, wherein a subsequent primitive from the plurality in the issue unit is processed if no dependency is detected between the subsequent primitive and the primitives in the at least two which have not yet been completely processed.

25.   (Currently amended) The apparatus of claim 23 wherein the issue unit ~~includes~~ comprises:

    source reservation stations for storing the source region bounding box coordinates for each primitive in the plurality stored in the issue unit; and

    destination reservation stations for storing the destination region bounding box coordinates for each primitive in the plurality stored in the issue unit.

26.   (Original) The apparatus of claim 25 wherein the comparators compare the destination region bounding box coordinates for the primitives in the at least two which have not yet been completely processed with the destination region bounding box coordinates and the source region bounding box coordinates for the next primitive to be processed in order to detect a dependency.

27   (Original) The apparatus of claim 26 wherein a dependency is detected if there is an overlap in the destination region bounding box coordinates for the primitives in the at least two which have not yet been completely processed and either the destination region bounding box coordinates or the source region bounding box coordinates for the next primitive to be processed.

28.   (Cancelled)

29.   (Currently Amended) ~~The method of claim 28,~~ A method for determining dependencies between a first graphics primitive and a second graphics primitive, the method comprising:

calculating a first bounding box for the first graphics primitive, the first bounding box surrounding a destination operand of the first graphics primitive;

calculating a second bounding box for the second graphic primitive, the second bounding box surrounding a destination operand of the second graphic primitive; and

determining whether the first bounding box and the second bounding box overlap, wherein a write after write dependency is detected if the bounding boxes overlap.